






Grupo: Los Inadaptados

Estudiantes:


- Camilo Lagos Malaver
- John Freddy Morenazo Alejo
- Juan Felipe Marin Moreno
- Sergio Nicolas Escobar

Código





- El lenguaje de programación: C#

Elegimos C# gracias a la confiabilidad  que tiene en el desarrollo de aplicaciones de escritorio, las mismas pueden incluso competir en velocidad  contra las escritas en C++, además Visual Studio proporciona diseñador visual , IntelliSense avanzado, depurador integrado, profiling y testing, lo que acelera enormemente el ciclo de desarrollo.



















- Framework seleccionado: .NET

.NET se destaca por su rendimiento y escalabilidad , su documentación completa, y la gran cantidad de librerías disponibles que tiene ([GitHub - quozd/awesome-dotnet: A collection of awesome .NET libraries, tools, frameworks and software](https://github.com/quozd/awesome-dotnet), algunos ejemplos).





- Comparativo de ventajas y desventajas con otras tecnologías

	Rendimiento	Compatibilidad	Versatilidad
C#	Alto rendimiento  en entornos Windows gracias a su integración con .NET y herramientas como WPF y	Excelente compatibilidad con Windows  , siendo el lenguaje nativo para aplicaciones de escritorio en este	Muy versátil: adecuado para aplicaciones de escritorio  , desarrollo web (ASP.NET)  , móviles (Xamarin)

	WinForms. Soporta programación asincrónica con async/await, facilitando la concurrencia 🔄.	sistema. Con .NET Core y .NET 5/6+ 🌐 ha mejorado su soporte multiplataforma, incluyendo macOS y Linux, aunque con algunas limitaciones en GUI.	📱, videojuegos (Unity) 🎮 y servicios en la nube ☁. Su ecosistema y herramientas de desarrollo son robustos, especialmente en entornos Microsoft.
Rust	Ofrece rendimiento comparable al de C++ ⚡, con ventajas en seguridad de memoria 🛡 y concurrencia sin necesidad de recolector de basura. Ideal para aplicaciones que requieren alta eficiencia y fiabilidad.	Aunque es multiplataforma 🌐, el desarrollo de interfaces gráficas en Windows aún está en evolución. Existen bibliotecas como druid e iced, pero no están tan maduras como las de otros lenguajes.	Principalmente utilizado en desarrollo de sistemas, software embebido y aplicaciones de alto rendimiento 🚀. Su ecosistema para aplicaciones de escritorio está creciendo, pero aún es limitado en comparación con otros lenguajes.
C++	Muy alto rendimiento 🚀 y control sobre recursos del sistema. Permite optimizaciones a bajo nivel 🔧, siendo ideal para aplicaciones que requieren máxima eficiencia.	Altamente compatible con Windows 🪟. Herramientas como MFC y bibliotecas como Qt y wxWidgets facilitan el desarrollo de aplicaciones de escritorio. Sin embargo, la gestión manual de memoria puede ser propensa a errores ⚠ si no se maneja cuidadosamente.	Ampliamente utilizado en desarrollo de videojuegos 🎮, sistemas embebidos ⚙ y aplicaciones científicas 🔬. Su versatilidad es alta, aunque la complejidad del lenguaje puede aumentar la curva de aprendizaje..
Java		Multiplataforma gracias a la JVM 🌐. En Windows,	Utilizado en aplicaciones empresariales 🏢,

	<p>Rendimiento moderado  debido a la ejecución en la JVM, aunque ha mejorado con optimizaciones como el JIT Compiler. Adecuado para aplicaciones empresariales y de tamaño medio .</p>	<p>se pueden desarrollar aplicaciones de escritorio utilizando bibliotecas como Swing y JavaFX. Sin embargo, la apariencia de las interfaces puede no integrarse completamente con el estilo nativo de Windows.</p>	<p>desarrollo web , aplicaciones móviles (Android)  y sistemas embebidos . Su versatilidad es amplia, respaldada por una gran comunidad y abundante documentación .</p>
Python	<p>Rendimiento inferior al de lenguajes compilados  debido a su naturaleza interpretada. Sin embargo, es suficiente para muchas aplicaciones de escritorio, especialmente aquellas que no requieren procesamiento intensivo.</p>	<p>Compatible con Windows y otras plataformas . Bibliotecas como Tkinter, PyQt y wxPython permiten el desarrollo de interfaces gráficas, aunque pueden no ofrecer el mismo nivel de integración nativa que otras herramientas.</p>	<p>Muy versátil: ampliamente utilizado en ciencia de datos , automatización , desarrollo web , scripting y aplicaciones de escritorio . Su sintaxis sencilla lo hace accesible para principiantes y eficiente para desarrolladores experimentados.</p>
JavaScript	<p>Rendimiento adecuado para aplicaciones de escritorio  cuando se utiliza con frameworks como Electron, que permiten crear aplicaciones multiplataforma  utilizando tecnologías web. Sin embargo,</p>	<p>Altamente compatible con Windows a través de Electron y similares. Permite desarrollar aplicaciones de escritorio utilizando HTML, CSS y JavaScript , aunque estas aplicaciones suelen ser más</p>	<p>Muy versátil en el desarrollo web , tanto en frontend como en backend (Node.js). También se utiliza en el desarrollo de aplicaciones móviles  y de escritorio  mediante frameworks que permiten reutilizar</p>

	estas aplicaciones pueden consumir más recursos ⚠ que las nativas.	pesadas en términos de consumo de memoria y recursos ⚠.	código web.
--	--	---	-------------

Framework	Tipo de Aplicación	Multiplataforma	Estado Actual	Ventajas	Desventajas
.NET Framework	Escritorio (WinForms, WPF)	 Solo Windows	En mantenimiento	Estable, compatible con proyectos legados, muchas bibliotecas	No multiplataforma, desactualizado para nuevos desarrollos
.NET (Core / 5+)	Escritorio, Web, Móvil (con MAUI, WPF, WinForms)	 Sí	Activo y recomendado	Plataforma moderna, unificada, mejor rendimiento	Algunas APIs aún migrando del .NET Framework tradicional
ASP.NET Core	Web, API REST, híbridas	 Sí	Activo y moderno	Ideal para aplicaciones web y APIs, backend de apps híbridas	No es para interfaces gráficas de escritorio directamente
Xamarin	Móvil, Escritorio limitado	 Sí	Sustituido por MAUI	Permite compartir lógica entre plataformas móviles usando C#	Complejo para escritorio, menos soporte que MAUI

.NET MAUI	Escritorio y Móvil (nativo)	✓ Sí	Activo, nuevo estándar	Reemplazo moderno de Xamarin, interfaz única multiplataforma	Aún en evolución, menor madurez que WPF o WinForms
Avalonia	Escritorio GUI	✓ Sí	Activo, open-source	Diseño moderno con XAML, multiplataforma real	Ecosistema pequeño, menor soporte corporativo
Uno Platform	Escritorio, Móvil, Web (WinUI)	✓ Sí	Activo	Reutiliza XAML de WinUI, exporta también a WebAssembly, Android, iOS	Curva de aprendizaje mayor, comunidad más reducida
Blazor Hybrid	Escritorio (web embebida)	✓ Sí	Activo y emergente	Combina C# con frontend web, útil con .NET MAUI	Uso intensivo de recursos, interfaz no nativa
Electron.NET	Escritorio con interfaz web (HTML/CSS/JS + C#)	✓ Sí	Activo	Usa ASP.NET Core con Electron, buena para apps tipo web en escritorio	Alto consumo de memoria, tamaño de app elevado
WPF (.NET)	Escritorio (GUI rica)	✗ Solo Windows	Activo (solo Windows)	Gran capacidad gráfica, soporte de MVVM,	Solo para Windows, no ideal para nuevos desarrollos

				potente en Windows	multiplatafor ma
WinForms (.NET)	Escritorio (GUI tradicional)	✗ Solo Windows	Activo (heredado)	Fácil de usar, desarrollo rápido para apps de negocio	Estética limitada, poca modernización, no multiplatafor ma

- La base de datos relacional que se utilizará: Postgres
- Bibliotecas y herramientas complementarias planeadas:

-Avalonia para la UI 🎨, se eligió gracias a lo ligero y modular que es, una amplitud de componentes disponibles 📦, y la posibilidad de poder crear fácilmente una UI moderna y personalizable ✨.

-Se utilizó Semi.Avalonia para el estilo de la aplicación 🎨, ya que tiene un estilo moderno y estético ✨, es fácil de integrar en un proyecto de Avalonia, es multiplataforma 🌐 y personalizable.

-NAudio para el audio 🎧, lo usamos ya que está escrita desde 0 en C#, pensada para .NET de manera nativa ✅. Más de una década de desarrollo le da una alta confiabilidad 🔧, una documentación abundante 📖, y un amplio soporte para distintos formatos de audio 🎵.